N 15

```
MMM         MMM     000000000     MMM         MMM
MMM         MMM     000000000     MMM         MMM
MMM         MMM     000000000     MMM         MMM
MMMMMM   MMMMMM  000         000  MMMMMM   MMMMMM
MMMMMM   MMMMMM  000         000  MMMMMM   MMMMMM
MMMMMM   MMMMMM  000         000  MMMMMM   MMMMMM
MMM   MMM   MMM  000         000  MMM   MMM   MMM
MMM   MMM   MMM  000         000  MMM   MMM   MMM
MMM   MMM   MMM  000         000  MMM   MMM   MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM     000000000     MMM         MMM
MMM         MMM     000000000     MMM         MMM
MMM         MMM     000000000     MMM         MMM
```

```
**FILE**ID**MOMSUBS                                     N 13
```

```
MM      MM    OOOOOO    MM      MM    SSSSSSSS  UU      UU  BBBBBBBB      SSSSSSSS
MM      MM    OOOOOO    MM      MM    SSSSSSSS  UU      UU  BBBBBBBB      SSSSSSSS
MMMM  MMMM  OO      OO  MMMM  MMMM  SS          UU      UU  BB      BB  SS
MMMM  MMMM  OO      OO  MMMM  MMMM  SS          UU      UU  BB      BB  SS
MM  MM  MM  OO      OO  MM  MM  MM  SS          UU      UU  BB      BB  SS
MM  MM  MM  OO      OO  MM  MM  MM  SS          UU      UU  BB      BB  SS
MM      MM  OO      OO  MM      MM    SSSSSS    UU      UU  BBBBBBBB      SSSSSS
MM      MM  OO      OO  MM      MM    SSSSSS    UU      UU  BBBBBBBB      SSSSSS
MM      MM  OO      OO  MM      MM          SS  UU      UU  BB      BB          SS
MM      MM  OO      OO  MM      MM          SS  UU      UU  BB      BB          SS
MM      MM  OO      OO  MM      MM          SS  UU      UU  BB      BB          SS      ....
MM      MM  OO      OO  MM      MM          SS  UU      UU  BB      BB          SS      ....
MM      MM    OOOOOO    MM      MM  SSSSSSSS    UUUUUUUUUU  BBBBBBBB    SSSSSSSS        ....
MM      MM    OOOOOO    MM      MM  SSSSSSSS    UUUUUUUUUU  BBBBBBBB    SSSSSSSS        ....

LL          IIIIII      SSSSSSSS
LL          IIIIII      SSSSSSSS
LL            II      SS
LL            II      SS
LL            II      SS
LL            II      SS
LL            II        SSSSSS
LL            II        SSSSSS
LL            II              SS
LL            II              SS
LL            II              SS
LL            II              SS
LLLLLLLLLL  IIIIII    SSSSSSSS
LLLLLLLLLL  IIIIII    SSSSSSSS
```

MOM
V04

```
0 %TITLE 'Special service routines'
0 MODULE MOMSUBS (
0                 LANGUAGE (BLISS32),
0                 ADDRESSING_MODE (NONEXTERNAL=GENERAL),
0                 ADDRESSING_MODE (EXTERNAL=GENERAL),
0                 IDENT = 'V04-000'
0                 ) =
1 BEGIN
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1 !
1 !
1 !++
1 ! FACILITY:  DECnet-VAX Network Maintenance Operations Module (MOM)
1 !
1 ! ABSTRACT:
1 !       This module contains utility routines used for maintenance operations.
1 !
1 ! ENVIRONMENT:  VAX/VMS Operating System
1 !
1 ! AUTHOR:  Kathy Perko
1 !
1 ! CREATION DATE:  6-Jan-1983
1 !
1 ! MODIFIED BY:
1 !       V03-005 MKP0005         Kathy Perko             26-June-1984
1 !               If sending a BOOT message for a LOAD command, set the bit
1 !               that tells the target to perform the load from this host.
1 !
1 !       V03-004 MKP0004         Kathy Perko             12-April-1984
1 !               Change padding on SERVICE PASSWORD to zero instead of high
1 !               byte.
1 !
1 !       V03-003 MKP0003         Kathy Perko             20-Jan-1984
1 !               Add SERVICE NODE VERSION parameter.
1 !               Pad the service password in the boot message with the
```

```
1 !                     high byte.
1 !
1 !             V03-002 MKP0002         Kathy Perko             23-May-1983
1 !                     When building the MOP Parameter Load with Transfer Address
1 !                     message, mask out the area number if the target isn't on the
1 !                     NI (this is a temporary way of identifying Phase III targets).
1 !
1 !             V03-001 MKP0001         Kathy Perko             11-May-1983
1 !                     Fix length of password put into MOP boot message.
1 !
1 !--
1
```

```
1 %SBTTL 'Declarations'
1
1 !
1 ! TABLE OF CONTENTS:
1 !
1
1 FORWARD ROUTINE
1     mom$getsrvdata         : NOVALUE,
1     mom$get_circuit_type: NOVALUE,
1     mom$get_node_id        : NOVALUE,
1     mom$getsrvtimer        : NOVALUE,
1     mom$get_voldb_data,
1     mom_get_circ_search2_key: NOVALUE,
1     mom$bldmoprds          : NOVALUE,
1     mom$bldmopboot         : NOVALUE,
1     mom$bldmopplt          : NOVALUE;
1
1 !
1 ! INCLUDE FILES:
1 !
1
1 LIBRARY 'LIB$:MOMLIB.L32';
1 LIBRARY 'SHRLIB$:NMALIBRY.L32';
1 LIBRARY 'SHRLIB$:NET.L32';
1 LIBRARY 'SYS$LIBRARY:STARLET.L32';
1
1 !
1 ! OWN STORAGE:
1 !
1
1 OWN
1     mom$t_p2buffer : VECTOR [mom$k_p2_buf_len, BYTE];    ! P2 QIO buffer
1
1 BIND
1     mom$q_p2_buf_dsc  = UPLIT (mom$k_p2_buf_len, mom$t_p2buffer) : VECTOR [2];
1
1 !
1 ! EXTERNAL REFERENCES:
1 !
1
1 $mom_externals;                              ! Define external service data
1
1 EXTERNAL
1     mom$npa_load,
1     mom$npa_cirloop,
1     mom$npa_trigger;
1
1 EXTERNAL ROUTINE
1     mom$bld_reply,
1     mom$build_p2,
1     mom$error,
1     mom$debug_msg,
1     mom$debug_txt,
1     mom$netacp_qio;
```

```
1 %SBTTL 'mom$getsrvdata  Build the service data base'
1 GLOBAL ROUTINE mom$getsrvdata : NOVALUE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine gets the information needed for a maintenance operation
1 !       from the target node's volatile data base entry.
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       Signal errors.
1 !
1 !--
1
2 BEGIN
2
2 LOCAL
2     datptr,
2     string_len,
2     p4_buf_dsc : VECTOR [2],
2     qio_p4_buffer : BBLOCK [mom$k_qio_buf_len];
2
2 IF .mom$gb_function NEQ nma$c_fnc_tes THEN
3     BEGIN
3     !
3     ! Get the maintenance parameters from NETACPs node database entry for the
3     ! target node.
3     !
3     p4_buf_dsc [0] = mom$k_qio_buf_len;
3     p4_buf_dsc [1] = qio_p4_buffer;
3
3     mom$get_voldb_data (nfb$c_db_ndi, p4_buf_dsc);
3     !
3     ! Build the service data table.  This table contains the values of longword
3     ! parameters, and pointers to string parameters.
3     !
3     datptr = qio_p4_buffer;
3     !
3     ! Some parameters have already been extracted from the NICE or MOP message
3     ! and inserted in the Service Data table.  These take precedence over
3     ! what's in the volatile database.  So, move the rest of the service
3     ! parameters from the QIOs P4 buffer into Service Data Table.
3     ! The field IDs were put into the NFB in the order they are in in the
3     ! Service Data Table.  Extract the parameter values from the P4 buffer
3     ! in the same order.
3     !
3     INCR i FROM 0 TO svd$c_entry_count DO
4         BEGIN
4         !
4         ! If the parameter value is obtained from the remote node (NDI)
4         ! database and it hasn't already been set by the NICE or MOP message,
4         ! put it into the Service Data Table.
4         !
4         IF .mom$ab_service_data [.i, svd$b_nfb_database]
4                                               EQL nfb$c_db_ndi THEN
```

```
5               BEGIN
5               IF .mom$ab_service_data [.i, svd$b_nice_type] NEQ
5                                                            svd$k_string THEN
5                    !
5                    ! If the parameter isn't a string and a value was returned
5                    ! for it, move its value into the Service Data Table.
5                    !
6                    BEGIN
6                    IF ..datptr GTR -1 AND
6                        NOT .mom$ab_service_data [.i, svd$v_msg_param] THEN
6                         mom$ab_service_data [.i, svd$l_param] = ..datptr;
6                    datptr = .datptr + 4;
6                    END
5               ELSE
5                    !
5                    ! If the parameter is a string, and a value was returned for
5                    ! it, move the string into Service Data Table.
5                    !
6                    BEGIN
6                    string_len = .(.datptr)<0,16>;
6                    IF .string_len GTR 0 AND
6                        NOT .mom$ab_service_data [.i, svd$v_msg_param] THEN
7                         BEGIN
7                         mom$ab_service_data [.i, svd$b_string_len] = .string_len;
7                         CH$MOVE (.string_len,
7                                   (.datptr + 2),
7                                     mom$ab_service_data [.i, svd$t_string]);
7                         END;
6                    datptr = .string_len + .datptr + 2;
5                    END;
4               END;
3          END;
3     !
3     ! Get the Host node id for Loads and dumps.
3     !
3     IF .mom$gb_function EQL nma$c_fnc_loa OR
3        .mom$gb_function EQL nma$c_fnc_dum THEN
3         mom$get_node_id (svd$gk_pcno_iho,
3                                svd$gk_pcno_$hna);
2     END;
2 !
2 ! Determine if service circuit is an NI circuit.  NI service operations
2 ! are different from point-to-point or multipoint at many points.  For
2 ! autoservice this is determined elsewhere.
2 !
2 IF NOT .mom$gl_service_flags [mom$v_autoservice] THEN
2    mom$get_circuit_type ();
1 END;                                    ! End of mom$getsrvdata


                                                   .TITLE  MOMSUBS Special service routines
                                                   .IDENT  \V04-000\

                                                   .PSECT  $PLIT$,NOWRT,NOEXE,2

                             00000068  00000 P.AAA:  .LONG   104                                     ;
```

```
                         00000000' 00004          .ADDRESS MOM$T_P2BUFFER                                     ;

                                                  .PSECT  $OWN$,NOEXE,2

                                   00000 MOM$T_P2BUFFER:
                                                  .BLKB   104

                                         MOM$Q_P2_BUF_DSC=    P.AAA
                                                  .EXTRN  MOM$GL_LOGMASK, MOM$GL_SVD_INDEX
                                                  .EXTRN  MOM$AB_SERVICE_DATA
                                                  .EXTRN  MOM$GB_FUNCTION
                                                  .EXTRN  MOM$GB_OPTION_BYTE
                                                  .EXTRN  MOM$GB_ENTITY_CODE
                                                  .EXTRN  MOM$AB_ENTITY_BUF
                                                  .EXTRN  MOM$GQ_ENTITY_BUF_DSC
                                                  .EXTRN  MOM$GL_SERVICE_FLAGS
                                                  .EXTRN  MOM$AB_NPARSE_BLK
                                                  .EXTRN  MOM$AB_NICE_RCV_BUF
                                                  .EXTRN  MOM$AB_NICE_XMIT_BUF
                                                  .EXTRN  MOM$GQ_NICE_RCV_BUF_DSC
                                                  .EXTRN  MOM$GL_NICE_RCV_MSG_LEN
                                                  .EXTRN  MOM$GQ_NICE_XMIT_BUF_DSC
                                                  .EXTRN  MOM$AB_MSGBLOCK
                                                  .EXTRN  MOM$AB_ACPQIO_BUFFER
                                                  .EXTRN  MOM$GQ_ACPQIO_BUF_DSC
                                                  .EXTRN  MOM$AB_CIB, MOM$AB_LOOP_CIB
                                                  .EXTRN  MOM$AB_TRIGGER_CIB
                                                  .EXTRN  MOM$AB_MOP_XMIT_BUF
                                                  .EXTRN  MOM$GQ_MOP_XMIT_BUF_DSC
                                                  .EXTRN  MOM$AB_MOP_RCV_BUF
                                                  .EXTRN  MOM$GQ_MOP_RCV_BUF_DSC
                                                  .EXTRN  MOM$AB_MOP_MSG, MOM$GQ_MOP_MSG_DSC
                                                  .EXTRN  MOM$GW_EVT_CODE
                                                  .EXTRN  MOM$GB_EVT_POPR
                                                  .EXTRN  MOM$GB_EVT_PRSN
                                                  .EXTRN  MOM$GB_EVT_PSER
                                                  .EXTRN  SVD$GK_PCNO_ADD
                                                  .EXTRN  SVD$GK_PCNO_SDV
                                                  .EXTRN  SVD$GK_PCNO_CPU
                                                  .EXTRN  SVD$GK_PCNO_STY
                                                  .EXTRN  SVD$GK_PCNO_DAD
                                                  .EXTRN  SVD$GK_PCNO_DCT
                                                  .EXTRN  SVD$GK_PCNO_IHO
                                                  .EXTRN  SVD$GK_PCNO_NNA
                                                  .EXTRN  SVD$GK_PCNO_SLI
                                                  .EXTRN  SVD$GK_PCNO_SPA
                                                  .EXTRN  SVD$GK_PCNO_HWA
                                                  .EXTRN  SVD$GK_PCNO_SNV
                                                  .EXTRN  SVD$GK_PCNO_LOA
                                                  .EXTRN  SVD$GK_PCNO_SLO
                                                  .EXTRN  SVD$GK_PCNO_TLO
                                                  .EXTRN  SVD$GK_PCNO_DFL
                                                  .EXTRN  SVD$GK_PCNO_SID
                                                  .EXTRN  SVD$GK_PCNO_DUM
                                                  .EXTRN  SVD$GK_PCNO_SDU
                                                  .EXTRN  SVD$GK_PCNO_$HNA
                                                  .EXTRN  SVD$GK_PCNO_$HHW
```

```
                                                                   H 14
MOMSUBS          Special service routines                          16-Sep-1984 02:08:44    VAX-11 Bliss-32 V4.0-742                Page   7
V04-000          mom$getsrvdata  Build the service data base       14-Sep-1984 12:44:37    DISK$VMSMASTER:[MOM.SRC]MOMSUBS.B32;1    (3)

                                                              .EXTRN   SVD$GK_PCNO_$FTY
                                                              .EXTRN   SVD$GK_PCNO_PHA
                                                              .EXTRN   SVD$GK_PCNO_$DA
                                                              .EXTRN   SVD$GK_PCNO_LPC
                                                              .EXTRN   SVD$GK_PCNO_LPL
                                                              .EXTRN   SVD$GK_PCNO_LPD
                                                              .EXTRN   SVD$GK_PCNO_LPH
                                                              .EXTRN   SVD$GK_PCNO_LPA
                                                              .EXTRN   SVD$GK_PCNO_LPN
                                                              .EXTRN   SVD$GK_PCNO_$LNA
                                                              .EXTRN   SVD$GK_PCNO_$LNH
                                                              .EXTRN   SVD$GK_PCNO_LAN
                                                              .EXTRN   SVD$GK_PCNO_$LNN
                                                              .EXTRN   SVD$GK_PCNO_$LAH
                                                              .EXTRN   SVD$GK_PCLI_STI
                                                              .EXTRN   SVD$C_ENTRY_COUNT
                                                              .EXTRN   MOM$NPA_LOAD, MOM$NPA_CIRLOOP
                                                              .EXTRN   MOM$NPA_TRIGGER
                                                              .EXTRN   MOM$BLD_REPLY, MOM$BUILD_P2
                                                              .EXTRN   MOM$ERROR, MOM$DEBUG_MSG
                                                              .EXTRN   MOM$DEBUG_TXT, MOM$NETACP_QIO

                                                              .PSECT   $CODE$,NOWRT,2

                                      07FC 00000              .ENTRY   MOM$GETSRVDATA, Save R2,R3,R4,R5,R6,R7,R8,-  ; 0125
                                                                       R9,R10                                       ;
                  5A 00000000G    00   9E 00002               MOVAB    MOM$GB_FUNCTION, R10                         ;
                  59 00000000G    00   9E 00009               MOVAB    MOM$AB_SERVICE_DATA+7, R9                    ;
                  5E        FDF8  CE   9E 00010               MOVAB    -520(SP), SP                                 ;
                  12              6A   91 00015               CMPB     MOM$GB_FUNCTION, #18                         ; 0148
                                  03   12 00018               BNEQ     1$                                           ;
                                008C   31 0001A               BRW      8$                                           ;
           F8     AD        0200  8F   3C 0001D 1$:           MOVZWL   #512, P4_BUF_DSC                             ; 0154
           FC     AD              6E   9E 00023               MOVAB    QIO_P4_BUFFER, P4_BUF_DSC+4                  ; 0155
                            F8    AD   9F 00027               PUSHAB   P4_BUF_DSC                                   ; 0157
                                  02   DD 0002A               PUSHL    #2                                           ;
        00000000V 00              02   FB 0002C               CALLS    #2, MOM$GET_VOLDB_DATA                       ;
                  57              6E   9E 00033               MOVAB    QIO_P4_BUFFER, DATPTR                        ; 0162
                  56              01   CE 00036               MNEGL    #1, I                                        ; 0172
                                  46   11 00039               BRB      6$                                           ;
    50            56 00000089     8F   C5 0003B 2$:           MULL3    #137, I, R0                                  ; 0179
                  02         FC A940   91 00043               CMPB     MOM$AB_SERVICE_DATA+3[R0], #2                ; 0180
                                  37   12 00048               BNEQ     6$                                           ;
                  03         FF A940   91 0004A               CMPB     MOM$AB_SERVICE_DATA+6[R0], #3                ; 0182
                                  15   13 0004F               BEQL     4$                                           ;
                                  67   D5 00051               TSTL     (DATPTR)                                     ; 0189
                                  0C   19 00053               BLSS     3$                                           ;
    07              6940          00   E0 00055               BBS      #0, MOM$AB_SERVICE_DATA+7[R0], 3$            ; 0190
                             02 A940   9F 0005A               PUSHAB   MOM$AB_SERVICE_DATA+9[R0]                    ; 0191
                  9E              67   D0 0005E               MOVL     (DATPTR), @(SP)+                             ;
                  57              04   C0 00061 3$:           ADDL2    #4, DATPTR                                   ; 0192
                                  1B   11 00064               BRB      6$                                           ; 0182
                  58              67   3C 00066 4$:           MOVZWL   (DATPTR), STRING_LEN                         ; 0200
                                  11   15 00069               BLEQ     5$                                           ; 0201
    0C              6940          00   E0 0006B               BBS      #0, MOM$AB_SERVICE_DATA+7[R0], 5$            ; 0202
               01 A940            58   90 00070               MOVB     STRING_LEN, MOM$AB_SERVICE_DATA+8[R0]        ; 0204
02 A940        02   A7            58   28 00075               MOVC3    STRING_LEN, 2(DATPTR), -                     ; 0207
```

```
                                                                          MOM$AB_SERVICE_DATA+9[R0]                       ;
                              57          02 A748   9E 0007C 5$:   MOVAB   2(DATPTR)[STRING_LEN], DATPTR                   ; 0209
              B2              56 00000000G     8F   F3 00081 6$:   AOBLEQ  #SVD$C_ENTRY_COUNT, I, 2$                       ; 0172
                              50               6A   9A 00089       MOVZBL  MOM$GB_FUNCTION, R0                             ; 0216
                              0F               50   91 0008C       CMPB    R0, #15                                         ;
                                               05   13 0008F       BEQL    7$                                              ;
                              10               50   91 00091       CMPB    R0, #16                                         ; 0217
                                               13   12 00094       BNEQ    8$                                              ;
                                 00000000G     8F   DD 00096 7$:   PUSHL   #SVD$GK_PCNO_$HNA                               ; 0218
                                 00000000G     8F   DD 0009C       PUSHL   #SVD$GK_PCNO_IHO                                ;
                   00000000V  00               02   FB 000A2       CALLS   #2, MOM$GET_NODE_ID                             ;
                              07 00000000G     00   E8 000A9 8$:   BLBS    MOM$GL_SERVICE_FLAGS, 9$                        ; 0227
                   00000000V  00               00   FB 000B0       CALLS   #0, MOM$GET_CIRCUIT_TYPE                        ; 0228
                                               04   000B7    9$:   RET                                                     ; 0229

; Routine Size:  184 bytes,    Routine Base:  $CODE$ + 0000
```

```
  1 %SBTTL 'mom$get_circuit_type   See if Circuit is on Ethernet'
  1 GLOBAL ROUTINE mom$get_circuit_type : NOVALUE =
  1
  1 !++
  1 ! FUNCTIONAL DESCRIPTION:
  1 !         This routine looks the service circuit up in the volatile database
  1 !         to determine if it's an NI circuit or not.
  1 !
  1 ! ROUTINE VALUE:
  1 ! COMPLETION CODES:
  1 !
  1 !         Signal errors.
  1 !
  1 !--
  1
  2 BEGIN
  2
P 2 $nfbdsc (mom_q_cirtyp_nfbdsc, show, , cri
P 2         ,nam,             ! Search key one = circuit name, oper  = eql
P 2         ,,                ! Null search key two.
P 2         ,typ              ! Circuit type
  2         );
  2
  2 LOCAL
  2     len,
  2     msgsize,
  2     p2dsc: VECTOR [2],
  2     p3,
  2     err_detail,
  2     status;
  2
  2 !
  2 ! If there isn't any service circuit for the node, return an error to NCP.
  2 ! (There is always a service circuit for autoservice functions).
  2 !
  2 len = .mom$ab_service_data [svd$gk_pcno_sli, svd$b_string_len];
  2 IF .len EQL 0 THEN
  3     BEGIN
  3     mom$ab_msgblock [msb$l_flags]  = msb$m_det_fld;
  3     mom$ab_msgblock [msb$b_code]   = nma$c_sts_pms;
  3     mom$ab_msgblock [msb$w_detail] = nma$c_pcno_sli;
  3     mom$bld_reply (mom$ab_msgblock, msgsize);
  3     $signal_msg (mom$ab_nice_xmit_buf, .msgsize);
  2     END;
  2 !
  2 ! Get the circuit type from NETACPs CRI database to determine if it's
  2 ! an NI (Ethernet) circuit.
  2 !
  2 mom$build_p2 (.len,
  2         mom$ab_service_data [svd$gk_pcno_sli, svd$t_string],
  2         -1, 0,
  2         mom$g_p2_buf_dsc, p2dsc);
  2 status = mom$netacp_qio (mom_q_cirtyp_nfbdsc,
  2                         p2dsc,
  2                         p3,
  2                         mom$gq_acpqio_buf_dsc);
  2 IF NOT .status THEN
```

```
3           BEGIN
3           mom$bld_reply (mom$ab_msgblock, msgsize);
3           $signal_msg (mom$ab_nice_xmit_buf, .msgsize);
2           END;
2       IF .(.mom$gq_acpqio_buf_dsc [1]) EQL nma$c_cirty_ni THEN
3           BEGIN
3           mom$gl_service_flags [mom$v_ni_circ] = true;
3           err_detail = 0;
3           !
3           ! If it's an NI circuit, and the NICE command was LOAD VIA, TRIGGER VIA,
3           ! it must also specify a physical address.  If it's LOOP CIRCUIT it must
3           ! specify a physical address or a node id.  This is because the circuit
3           ! id is not sufficient to uniquely identify a target on the NI.
3           !
3           IF NOT .mom$gl_service_flags [mom$v_autoservice] AND
3               NOT .mom$ab_service_data [svd$gk_pcno_pha, svd$v_msg_param] THEN
4               BEGIN
4               IF .mom$gb_entity_code EQL mom$c_circuit THEN
5                   BEGIN
5                   IF .mom$gb_function NEQ nma$c_fnc_tes AND
5                       NOT .mom$ab_service_data [svd$gk_pcno_add, svd$v_msg_param] AND
5                       NOT .mom$ab_service_data [svd$gk_pcno_nna, svd$v_msg_param] THEN
5                         err_detail = nma$c_pcno_pha
5                   ELSE
5                   IF NOT .mom$ab_service_data [svd$gk_pcno_lpn, svd$v_msg_param] AND
5                       NOT .mom$ab_service_data [svd$gk_pcno_$lna, svd$v_msg_param] AND
5                       NOT .mom$ab_service_data [svd$gk_pcno_lan, svd$v_msg_param] AND
5                       NOT .mom$ab_service_data [svd$gk_pcno_$lnn, svd$v_msg_param] THEN
5                         err_detail = nma$c_pcno_pha;
5                   END
4               ELSE
4                   !
4                   ! If it's an NI circuit, and the NICE command was LOAD NODE or
4                   ! TRIGGER NODE with no PHYSICAL ADDRESS specified, there must
4                   ! be a hardware address in the volatile database.
4                   !
5                   BEGIN
5                   IF .mom$ab_service_data [svd$gk_pcno_hwa, svd$b_string_len]
5                                                                       EQL 0 THEN
5                         err_detail = nma$c_pcno_hwa;
4                   END;
4               IF .err_detail NEQ 0 THEN
5                   BEGIN
5                   mom$ab_msgblock [msb$l_flags]  = msb$m_det_fld;
5                   mom$ab_msgblock [msb$b_code]   = nma$c_sts_pms;
5                   mom$ab_msgblock [msb$w_detail] = .err_detail;
5                   mom$bld_reply (mom$ab_msgblock, msgsize);
5                   $signal_msg (mom$ab_nice_xmit_buf, .msgsize);
4                   END;
3               END;
2           END;
1       END;                            ! of mom$get_circuit_type


                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2
```

```
                                                          0000001C 00008 P.AAB:   .LONG   28                                          ;
                                                          00000000' 0000C         .ADDRESS U.1                                        ;

                                                                                  .PSECT  $OWN$,NOEXE,2

                                                                22 00068 ; NFB
                                                                         U.1:     .BYTE   34                                          ;
                                                                00 00069          .BYTE   0                                           ;
                                                                04 0006A          .BYTE   4                                           ;
                                                                00 0006B          .BYTE   0                                           ;
                                                          04020041 0006C          .LONG   67240001                                    ;
                                                          00000001 00070          .LONG   1                                           ;
                                                                00 00074          .BYTE   0                                           ;
                                                                00 00075          .BYTE   0                                           ;
                                                              0000 00076          .WORD   0                                           ;
                                                          04010020 00078          .LONG   67174432                                    ;
                                                          00000000 0007C          .LONG   0                                           ;
                                                                   00080          .BLKB   4

                                                                         U.2=                 P.AAB


                                                                                  .PSECT  $CODE$,NOWRT,2

                                                              00FC 00000          .ENTRY  MOM$GET_CIRCUIT_TYPE, Save R2,R3,R4,R5,R6,-   ; 0231
                                                                                          R7
                                             57  00000000G  00  9E 00002          MOVAB   MOM$GL_SERVICE_FLAGS, R7                    ;
                                             56  00000000G  00  9E 00009          MOVAB   LIB$SIGNAL, R6                              ;
                                             55  00000000G  00  9E 00010          MOVAB   MOM$AB_NICE_XMIT_BUF, R5                    ;
                                             54  00000000G  00  9E 00017          MOVAB   MOM$BLD_REPLY, R4                           ;
                                             53  00000000G  00  9E 0001E          MOVAB   MOM$AB_MSGBLOCK, R3                         ;
                                             5E             10  C2 00025          SUBL2   #16, SP                                     ;
                                             52  00000000*  00  9A 00028          MOVZBL  <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_SLI*137>-  ; 0265
                                                                                          >+8>, LEN
                                                            22  12 0002F          BNEQ    1$                                          ; 0266
                                             63             02  D0 00031          MOVL    #2, MOM$AB_MSGBLOCK                         ; 0268
                                        04   A3             1D  8E 00034          MNEGB   #29, MOM$AB_MSGBLOCK+4                      ; 0269
                                        08   A3         6E  8F  9B 00038          MOVZBW  #110, MOM$AB_MSGBLOCK+8                     ; 0270
                                                        04  AE  9F 0003D          PUSHAB  MSGSIZE                                     ; 0271
                                                            53  DD 00040          PUSHL   R3                                          ;
                                             64             02  FB 00042          CALLS   #2, MOM$BLD_REPLY                           ;
                                                        04  AE  DD 00045          PUSHL   MSGSIZE                                     ; 0272
                                                            55  DD 00048          PUSHL   R5                                          ;
                                                 02070000   8F  DD 0004A          PUSHL   #34013184                                   ;
                                             66             03  FB 00050          CALLS   #3, LIB$SIGNAL                              ;
                                                        08  AE  9F 00053 1$:      PUSHAB  P2DSC                                       ; 0279
                                                 00000000'  00  9F 00056          PUSHAB  MOM$Q_P2_BUF_DSC                            ;
                                                            7E  D4 0005C          CLRL    -(SP)                                       ;
                                             7E             01  CE 0005E          MNEGL   #1, -(SP)                                   ; 0280
                                                 00000000*  00  9F 00061          PUSHAB  <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_SLI*137>-  ; 0279
                                                                                          >+9>
                                                            52  DD 00067          PUSHL   LEN                                         ;
                                  00000000G  00             06  FB 00069          CALLS   #6, MOM$BUILD_P2                            ;
                                                 00000000G  00  9F 00070          PUSHAB  MOM$GQ_ACPQIO_BUF_DSC                       ; 0282
                                                        04  AE  9F 00076          PUSHAB  P3                                          ;
                                                        10  AE  9F 00079          PUSHAB  P2DSC                                       ;
                                                 00000000'  00  9F 0007C          PUSHAB  U.2                                         ;
```

```
MOMSUBS         Special service routines                      M 14
                                                     16-Sep-1984 02:08:44    VAX-11 Bliss-32 V4.0-742              Page  12
V04-000         mom$get_circuit_type   See if Circuit is on Eth 14-Sep-1984 12:44:37    DISK$VMSMASTER:[MOM.SRC]MOMSUBS.B32;1   (4)

          00000000G  00              04  FB  00082        CALLS   #4, MOM$NETACP_QIO                              ;
                     16              50  E8  00089        BLBS    STATUS, 2$                                      ; 0286
                             04      AE  9F  0008C        PUSHAB  MSGSIZE                                         ; 0288
                                     53  DD  0008F        PUSHL   R3                                              ;
                     64              02  FB  00091        CALLS   #2, MOM$BLD_REPLY                               ;
                             04      AE  DD  00094        PUSHL   MSGSIZE                                         ; 0289
                                     55  DD  00097        PUSHL   R5                                              ;
                       02070000      8F  DD  00099        PUSHL   #34013184                                       ;
                     66              03  FB  0009F        CALLS   #3, LIB$SIGNAL                                  ;
                     50 00000000G    00  D0  000A2  2$:   MOVL    MOM$GQ_ACPQIO_BUF_DSC+4, R0                     ; 0291
                     06              60  D1  000A9        CMPL    (R0), #6                                        ;
                                     01  13  000AC        BEQL    3$                                              ;
                                         04  000AE        RET                                                     ;
                     67              02  88  000AF  3$:   BISB2   #2, MOM$GL_SERVICE_FLAGS                        ; 0293
                                     50  D4  000B2        CLRL    ERR_DETAIL                                      ; 0294
                     79              67  E8  000B4        BLBS    MOM$GL_SERVICE_FLAGS, 8$                        ; 0301
                     72 00000000*    00  E8  000B7        BLBS    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_PHA*137>-; 0302
                                                                  >+7>, 8$
                     02 00000000G    00  91  000BE        CMPB    MOM$GB_ENTITY_CODE, #2                          ; 0304
                                     38  12  000C5        BNEQ    6$                                              ;
                     12 00000000G    00  91  000C7        CMPB    MOM$GB_FUNCTION, #18                            ; 0306
                                     0E  13  000CE        BEQL    4$                                              ;
                     07 00000000*    00  E8  000D0        BLBS    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_ADD*137>-; 0307
                                                                  >+7>, 4$
                     1C 00000000*    00  E9  000D7        BLBC    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_NNA*137>-; 0308
                                                                  >+7>, 5$
                     26 00000000*    00  E8  000DE  4$:   BLBS    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPN*137>-; 0311
                                                                  >+7>, 7$
                     1F 00000000*    00  E8  000E5        BLBS    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_$LNA*-     ; 0312
                                                                  137>>+7>, 7$
                     18 00000000*    00  E8  000EC        BLBS    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LAN*137>-; 0313
                                                                  >+7>, 7$
                     11 00000000*    00  E8  000F3        BLBS    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_$LNN*-     ; 0314
                                                                  137>>+7>, 7$
                     50              0A  D0  000FA  5$:   MOVL    #10, ERR_DETAIL                                 ; 0315
                                     0C  11  000FD        BRB     7$                                              ; 0304
                        00000000*    00  95  000FF  6$:   TSTB    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_HWA*137>-; 0325
                                                                  >+8>
                                     04  12  00105        BNEQ    7$                                              ;
                     50      72      8F  9A  00107        MOVZBL  #114, ERR_DETAIL                                ; 0326
                                     50  D5  0010B  7$:   TSTL    ERR_DETAIL                                      ; 0328
                                     21  13  0010D        BEQL    8$                                              ;
                     63              02  D0  0010F        MOVL    #2, MOM$AB_MSGBLOCK                             ; 0330
                  04 A3              1D  8E  00112        MNEGB   #29, MOM$AB_MSGBLOCK+4                          ; 0331
                  08 A3              50  B0  00116        MOVW    ERR_DETAIL, MOM$AB_MSGBLOCK+8                   ; 0332
                             04      AE  9F  0011A        PUSHAB  MSGSIZE                                         ; 0333
                                     53  DD  0011D        PUSHL   R3                                              ;
                     64              02  FB  0011F        CALLS   #2, MOM$BLD_REPLY                               ;
                             04      AE  DD  00122        PUSHL   MSGSIZE                                         ; 0334
                                     55  DD  00125        PUSHL   R5                                              ;
                       02070000      8F  DD  00127        PUSHL   #34013184                                       ;
                     66              03  FB  0012D        CALLS   #3, LIB$SIGNAL                                  ;
                                         04  00130  8$:   RET                                                     ; 0338

; Routine Size:  305 bytes,    Routine Base:  $CODE$ + 00B8
```

```
%SBTTL 'mom$get_node_id  Get the name of the host node'
GLOBAL ROUTINE mom$get_node_id (node_add_svd,
                                node_name_svd,
                                NI_hwa_svd) : NOVALUE =

!++
! FUNCTIONAL DESCRIPTION:
!       This routine gets the node name and node address needed for a
!       load, dump, or loop circuit operation.  It uses the SVD indices
!       to determine what node name or address is already known (from
!       the NICE command or the volatile database), and gets the node
!       name, address, and NI hardware address for that node.  If no
!       node name or address is already known, the executor node is used.
!
! FORMAL PARAMETERS:
!       NODE_ADD_SVD    = Service Data (SVD) table index of entry for node
!                         address.
!       NODE_NAME_SVD   = Service Data (SVD) table index of entry for node
!                         name.
!       NI_HWA_SVD      = Service Data (SVD) table indes of NI hardware
!                         address for node.  Set up only for loop functions.
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       Signal errors.
!
!--

BEGIN

$nfbdsc(nfbdsc, show, , ndi
        ,add,             ! Search key 1 = node address, oper1 = eql
        ,nfb$c_wildcard,! Search key 2 = wildcard, oper2 = eql
        ,tad              ! Node address
        ,nna              ! Node name
        ,hwa);            ! NI hardware address

MAP
    nfbdsc:     VECTOR;

LOCAL
    search_key,
    search_len,
    sear.h_value,
    status,
    p2_dsc:     VECTOR [2],
    p2_buf_dsc: VECTOR [2],
    p2_buffer:  BBLOCK [mom$k_p2_buf_len],
    nfb:        REF BBLOCK,
    p4_dsc:     VECTOR [2],
    p4_buffer:  BBLOCK [32],
    ptr,
    length;


!
```

```
2 ! If the node name was supplied in the NICE command, use it to get the address.
2 !
2 IF .mom$ab_service_data [.node_name_svd, svd$v_msg_param] THEN
3     BEGIN
3     search_len = .mom$ab_service_data [.node_name_svd, svd$b_string_len];
3     search_value = mom$ab_service_data [.node_name_svd, svd$t_string];
3     search_key = nfb$c_ndi_nna;
3     END
2 ELSE
3     BEGIN
3     search_key = nfb$c_ndi_tad;
3     search_len = 0;
3     !
3     ! If the node address was supplied in the NICE command, use it to get
3     ! the name.  Otherwise, get the executor's name and address (this works
3     ! because the SVD$L_PARAM is initialized to 0).
3     !
3     search_value = .mom$ab_service_data [.node_add_svd, svd$l_param];
2     END;
2 !
2 ! Get the name and address of the node from the volatile data base.
2 ! If it is not found then report an error in the node identification parameter.
2 !
2 p2_buf_dsc [0] = mom$k_p2_buf_len;
2 p2_buf_dsc [1] = p2_buffer;
2 mom$build_p2 (.search_len,
2               .search_value,
2               -1, 0,
2               p2_buf_dsc, p2_dsc);
2 nfb = .nfbdsc [1];
2 nfb [nfb$l_srch_key] = .search_key;
2 p4_dsc [0] = 32;
2 p4_dsc [1] = p4_buffer;
2 IF mom$netacp_qio (      nfbdsc,
2                         p2_dsc,
2                         0,
2                         p4_dsc) THEN
3     BEGIN
3     ptr = p4_buffer;
3     !
3     ! If the node name and/or address were not supplied in the NICE command,
3     ! take the ones returned from the volatile database, and put them into
3     ! the service data.
3     !
3     IF NOT .mom$ab_service_data [.node_add_svd, svd$v_msg_param] THEN
3         mom$ab_service_data [.node_add_svd, svd$l_param] = ..ptr;
3     ptr = .ptr + 4;
3     length = .(.ptr)<0,16>;
3     IF NOT .mom$ab_service_data [.node_name_svd, svd$v_msg_param] THEN
4         BEGIN
4         CH$MOVE (.length, (.ptr + 2),
4                 mom$ab_service_data [.node_name_svd, svd$t_string]);
4         mom$ab_service_data [.node_name_svd, svd$b_string_len] = .length;
3         END;
3     ptr = .ptr + 2 + .length;
3     !
3     ! If it's a LOOP CIRCUIT function, also return the NI hardware address.
```

```
;  457          0453  3         ! For LOAD, TRIGGER, and DUMP functions, the hardware address is obtained
;  458          0454  3         ! with the rest of the service data.
;  459          0455  3         !
;  460          0456  3         IF .mom$gb_function EQL nma$c_fnc_tes THEN
;  461          0457  4             BEGIN
;  462          0458  4             length = .(.ptr)<0,16>;
;  463          0459  4             CH$MOVE (.length, (.ptr + 2),
;  464          0460  4                     mom$ab_service_data [.ni_hwa_svd, svd$t_string]);
;  465          0461  4             mom$ab_service_data [.ni_hwa_svd, svd$b_string_len] = .length;
;  466          0462  3             END;
;  467          0463  3         END
;  468          0464  2     ELSE
;  469          0465  2         mom$error (nma$c_sts_ide, nma$c_ent_nod);
;  470          0466  2
;  471          0467  1     END;                                ! End of mom$get_node_id



                                                         .PSECT  $PLIT$,NOWRT,NOEXE,2

                               00000024  00010 P.AAC:   .LONG   36                              ;
                               00000000' 00014          .ADDRESS U.3                            ;

                                                         .PSECT  $OWN$,NOEXE,2

                                     22  00084 ;_NFB
                                           U.3:         .BYTE   34                              ;
                                     00  00085          .BYTE   0                               ;
                                     02  00086          .BYTE   2                               ;
                                     00  00087          .BYTE   0                               ;
                               02010012  00088          .LONG   33619986                        ;
                               00000001  0008C          .LONG   1                               ;
                                     00  00090          .BYTE   0                               ;
                                     00  00091          .BYTE   0                               ;
                                   0000  00092          .WORD   0                               ;
                               02010010  00094          .LONG   33619984                        ;
                               02020043  00098          .LONG   33685571                        ;
                               02020057  0009C          .LONG   33685591                        ;
                               00000000  000A0          .LONG   0                               ;
                                         000A4          .BLKB   4

                                           U.4=                  P.AAC


                                                         .PSECT  $CODE$,NOWRT,2

                                   03FC  00000          .ENTRY  MOM$GET_NODE_ID, Save R2,R3,R4,R5,R6,R7,R8,-;  0340
                                                                R9
                  59 00000000G    00  9E 00002          MOVAB   MOM$AB_SERVICE_DATA+9, R9           ;
                  5E       FF60   CE  9E 00009          MOVAB   -160(SP), SP                        ;
    56      08    AC 00000089     8F  C5 0000E          MULL3   #137, NODE_NAME_SVD, R6             ;  0398
                  54         FE A946  9E 00017          MOVAB   MOM$AB_SERVICE_DATA+7[R6], R4       ;
                  12              64  E9 0001C          BLBC    (R4), T$                            ;
                  52         FF A946  9A 0001F          MOVZBL  MOM$AB_SERVICE_DATA+8[R6], SEARCH_LEN  ;  0400
    51            56              59  C1 00024          ADDL3   R9, R6, SEARCH_VALUE                ;  0401
                  53 02020043     8F  D0 00028          MOVL    #33685571, SEARCH_KEY               ;  0402
                                  18  11 0002F          BRB     2$                                  ;  0398
```

```
                        53 02010010     8F  D0 00031 1$:    MOVL    #33619984, SEARCH_KEY                       ; 0406
                                        52  D4 00038        CLRL    SEARCH_LEN                                  ; 0407
          50      04    AC 00000089     8F  C5 0003A        MULL3   #137, NODE_ADD_SVD, R0                      ; 0413
                                      6940  9F 00043        PUSHAB  MOM$AB_SERVICE_DATA+9[R0]                   ;
                        51              9E  D0 00046        MOVL    @(SP)+, SEARCH_VALUE                        ;
                  F0    AD        68    8F  9A 00049 2$:    MOVZBL  #104, P2_BUF_DSC                            ; 0419
                  F4    AD        28    AE  9E 0004E        MOVAB   P2_BUFFER, P2_BUF_DSC+4                     ; 0420
                                  F8    AD  9F 00053        PUSHAB  P2_DSC                                      ; 0421
                                  F0    AD  9F 00056        PUSHAB  P2_BUF_DSC                                  ;
                                        7E  D4 00059        CLRL    -(SP)                                       ;
                        7E              01  CE 0005B        MNEGL   #1, -(SP)                                   ; 0423
                                        51  DD 0005E        PUSHL   SEARCH_VALUE                                ; 0422
                                        52  DD 00060        PUSHL   SEARCH_LEN                                  ; 0421
             00000000G  00              06  FB 00062        CALLS   #6, MOM$BUILD_P2                            ;
                        50 00000000'    00  D0 00069        MOVL    NFBDSC+4, NFB                               ; 0425
                  04    A0              53  D0 00070        MOVL    SEARCH_KEY, 4(NFB)                          ; 0426
                  20    AE              20  D0 00074        MOVL    #32, P4_DSC                                 ; 0427
                  24    AE              6E  9E 00078        MOVAB   P4_BUFFER, P4_DSC+4                         ; 0428
                                  20    AE  9F 0007C        PUSHAB  P4_DSC                                      ; 0429
                                        7E  D4 0007F        CLRL    -(SP)                                       ;
                                  F8    AD  9F 00081        PUSHAB  P2_DSC                                      ;
                           00000000'    00  9F 00084        PUSHAB  NFBDSC                                      ;
             00000000G  00              04  FB 0008A        CALLS   #4, MOM$NETACP_QIO                          ;
                        52              50  E9 00091        BLBC    R0, 5$                                      ;
                        57              6E  9E 00094        MOVAB   P4_BUFFER, PTR                              ; 0434
          50      04    AC 00000089     8F  C5 00097        MULL3   #137, NODE_ADD_SVD, R0                      ; 0440
          06      FE  A940              00  E0 000A0        BBS     #0, MOM$AB_SERVICE_DATA+7[R0], 3$           ;
                                      6940  9F 000A6        PUSHAB  MOM$AB_SERVICE_DATA+9[R0]                   ; 0441
                        9E              67  D0 000A9        MOVL    (PTR), @(SP)+                               ;
                        57              04  C0 000AC 3$:    ADDL2   #4, PTR                                     ; 0442
                        58              67  3C 000AF        MOVZWL  (PTR), LENGTH                               ; 0443
                        0B              64  E8 000B2        BLBS    (R4), 4$                                    ; 0444
        6946      02    A7              58  28 000B5        MOVC3   LENGTH, 2(PTR), MOM$AB_SERVICE_DATA+9[R6]   ; 0447
                  FF  A946              58  90 000BB        MOVB    LENGTH, MOM$AB_SERVICE_DATA+8[R6]           ; 0448
                        57        02  A847  9E 000C0 4$:    MOVAB   2(LENGTH)[PTR], PTR                         ; 0450
                        12 00000000G    00  91 000C5        CMPB    MOM$GB_FUNCTION, #18                        ; 0456
                                        24  12 000CC        BNEQ    6$                                          ;
                        58              67  3C 000CE        MOVZWL  (PTR), LENGTH                               ; 0458
          56      0C    AC 00000089     8F  C5 000D1        MULL3   #137, NI_HWA_SVD, R6                        ; 0460
        6946      02    A7              58  28 000DA        MOVC3   LENGTH, 2(PTR), MOM$AB_SERVICE_DATA+9[R6]   ;
                  FF  A946              58  90 000E0        MOVB    LENGTH, MOM$AB_SERVICE_DATA+8[R6]           ; 0461
                                        04  000E5           RET                                                 ; 0429
                                        7E  D4 000E6 5$:    CLRL    -(SP)                                       ; 0465
                        7E              09  CE 000E8        MNEGL   #9, -(SP)                                   ;
             00000000G  00              02  FB 000EB        CALLS   #2, MOM$ERROR                               ;
                                        04  000F2 6$:       RET                                                 ; 0467

; Routine Size:  243 bytes,    Routine Base:  $CODE$ + 01E9


;  472          0468  1
;  474          0469  1 %SBTTL 'mom$getsrvtimer  Get the service timer'
;  475          0470  1 GLOBAL ROUTINE mom$getsrvtimer: NOVALUE =
;  476          0471  1
;  477          0472  1 !++
;  478          0473  1 ! FUNCTIONAL DESCRIPTION:
;  479          0474  1 !
;  480          0475  1 !       This routine gets the service timer of the circuit to be used.
;  481          0476  1 !       Since service timer is a line parameter, the routine must access
;  482          0477  1 !       the volatile data base of the line which corresponds to the
;  483          0478  1 !       target node's service circuit.
;  484          0479  1 !
;  485          0480  1 ! FORMAL PARAMETERS:
;  486          0481  1 !
;  487          0482  1 ! IMPLICIT INPUTS:
;  488          0483  1 !       Service Data Table (MOM$AB_SERVICE_DATA)
;  489          0484  1 !
;  490          0485  1 ! ROUTINE VALUE:
;  491          0486  1 ! COMPLETION CODES:
;  492          0487  1 !
;  493          0488  1 !       Signal errors.
;  494          0489  1 !
;  495          0490  1 !--
;  496          0491  1
;  497          0492  2 BEGIN
;  498          0493  2
;  499          0494  2 LOCAL
;  500          0495  2     p4_buf_dsc : VECTOR [2],
;  501          0496  2     qio_p4_buffer : BBLOCK [mom$k_qio_buf_len],
;  502          0497  2     status;
;  503          0498  2
;  504          0499  2 !
;  505          0500  2 ! Get the maintenance parameters from NETACPs node database entry for the
;  506          0501  2 ! target node.
;  507          0502  2 !
;  508          0503  2 p4_buf_dsc [0] = mom$k_qio_buf_len;
;  509          0504  2 p4_buf_dsc [1] = qio_p4_buffer;
;  510          0505  2
;  511          0506  2 status = mom$get_voldb_data (nfb$c_db_pli, p4_buf_dsc);
;  512          0507  2 IF .status THEN
;  513          0508  2 !
;  514          0509  2 ! Return the service timer value.  If the parameter is not set then
;  515          0510  2 ! the value will be -1.  This is a suitable value for infinity.
;  516          0511  2 ! Note that the service timer is defaulted to -1 when MOM is initializing.
;  517          0512  2 !
;  518          0513  2     mom$ab_service_data [svd$gk_pcli_sti, svd$l_param] = .qio_p4_buffer;
;  519          0514  1 END;                               ! End of mom$getsrvtimer



                                            0000 00000          .ENTRY  MOM$GETSRVTIMER, Save nothing              ; 0470
                      5E       FDF8    CE    9E 00002          MOVAB   -520(SP), SP                               ;
                 F8   AD       0200    8F    3C 00007          MOVZWL  #512, P4_BUF_DSC                           ; 0503
                 FC   AD               6E    9E 0000D          MOVAB   QIO_P4_BUFFER, P4_BUF_DSC+4                ; 0504
                               F8      AD    9F 00011          PUSHAB  P4_BUF_DSC                                 ; 0506
                                       05    DD 00014          PUSHL   #5                                         ;
```

```
                        00000000V  00                02  FB 00016          CALLS   #2, MOM$GET_VOLDB_DATA                          ;
                                   07                50  E9 0001D          BLBC    STATUS, 1$                                      ; 0507
                        00000000*  00                6E  D0 00020          MOVL    QIO_P4_BUFFER, <<MOM$AB_SERVICE_DATA+-          ; 0513
                                                                                   <SVD$GR_PCLI_STI*137>>+9>                       ;
                                                         04 00027 1$:      RET                                                     ; 0514

; Routine Size:  40 bytes,    Routine Base:  $CODE$ + 02DC


;  520          0515  1
```

```
1 %SBTTL 'mom$get_voldb_data  Get data from volatile database'
1 GLOBAL ROUTINE mom$get_voldb_data (database, p4_buf_dsc) : =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !       This routine builds the QIO buffers to get information about the
1 !       target from the volatile data base specified.  It issues the
1 !       QIO to NETACP.
1 !
1 ! Inputs:
1 !       DATABASE - Database id to use when building the NFB and to determine
1 !               which of the parameters in the Service Data Table
1 !               to request.
1 !       P4_BUF_DSC - P4 buffer descriptor in which to return information.
1 !
1 ! IMPLICIT INPUTS:
1 !       MOM$GB_ENTITY_CODE
1 !       MOM$GQ_ENTITY_BUF_DSC
1 !
1 ! OUTPUTS:
1 !       The P4 buffer described by P4_BUF_DSC contains the maintenance
1 !       information from the specified database.
1 !
1 !--
1
2 BEGIN
2
2 MAP
2     p4_buf_dsc : REF VECTOR;
2
2 LOCAL
2     status,
2     p2_dsc : VECTOR [2],
2     key,
2     length,
2     address,
2     line_len,
2     period_ptr,
2     nfb : REF BBLOCK,
2     nfbdsc : VECTOR [2],
2     nfb_buffer : BBLOCK [mom$k_qio_buf_len],
2     msgsize;
2
2 !
2 ! Build the NFB, which tells NETACP which information you want returned.
2 !
2 CH$FILL (0, mom$k_qio_buf_len, nfb_buffer);
2 nfb = nfb_buffer;
2 nfb [nfb$b_fct] = nfb$c_fc_show;
2 nfb [nfb$b_database] = .database;
2 nfb [nfb$b_oper] = nfb$c_op_eql;
2 nfb [nfb$l_srch2_key] = nfb$c_wildcard;
2 nfb [nfb$b_oper2] = nfb$c_op_eql;
2
2 !
2 ! Build the P2 buffer for the specified entity.  The P2 buffer identifies
2 ! the specific circuit or node for which information is being requested.
```

```
!
SELECTONEU .database OF
    SET
    [nfb$c_db_ndi]:
        SELECTONEU .mom$gb_entity_code OF
            SET

            [mom$c_circuit]:
                BEGIN
                nfb [nfb$l_srch_key] = nfb$c_ndi_sli;
                !
                ! Figure out what the second search key should be.  It's
                ! either the node address or the hardware address, depending
                ! on whether the physical address is the UNA hardware address
                ! or the hiord (node address with DEC NI address space constant)
                ! address.
                !
                mom_get_circ_search2_key (key, length, address);
                nfb [nfb$l_srch2_key] = .key;
                mom$build_p2 (  .mom$gq_entity_buf_dsc [0],
                                .mom$gq_entity_buf_dsc [1],
                                .length, .address,
                                mom$q_p2_buf_dsc, p2_dsc);
                END;

            [mom$c_node]:
                BEGIN
                nfb [nfb$l_srch_key] = nfb$c_ndi_add;
                mom$build_p2 (  0,
                                .(.mom$gq_entity_buf_dsc [1])<0,16>,
                                -1, 0,
                                mom$q_p2_buf_dsc, p2_dsc);
                END;

            [mom$c_nodebyname]:
                BEGIN
                nfb [nfb$l_srch_key] = nfb$c_ndi_nna;
                mom$build_p2 (  .mom$gq_entity_buf_dsc [0],
                                .mom$gq_entity_buf_dsc [1],
                                -1, 0,
                                mom$q_p2_buf_dsc, p2_dsc);
                END;
            TES;

    [nfb$c_db_pli]:
        BEGIN
        nfb [nfb$l_srch_key] = nfb$c_pli_nam;
        !
        ! If the service circuit for the target node is multidrop (eg. DMP-0.1),
        ! the corresponding line name will include the period and tributary
        ! number.  If so, before using the circuit name to access the ACPs line
        ! database, eliminate the period and tributary number from the end of the
        ! circuit name to get the line name.
        !
        line_len = .mom$ab_service_data [svd$gk_pcno_sli, svd$b_string_len];
        period_ptr = CH$FIND_CH (.line_len,
                        mom$ab_service_data [svd$gk_pcno_sli, svd$t_string],
;  636          0630  3                         %C'.');
;  637          0631  3             IF NOT CH$FAIL (.period_ptr) THEN
;  638          0632  3                 line_len = .period_ptr - mom$ab_service_data [svd$gk_pcno_sli,
;  639          0633  3                                                             svd$t_string];
;  640          0634  3
;  641          0635  3             mom$build_p2 (.line_len,
;  642          0636  3                         mom$ab_service_data [svd$gk_pcno_sli, svd$t_string],
;  643          0637  3                         -1, 0,
;  644          0638  3                         mom$q_p2_buf_dsc, p2_dsc);
;  645          0639  2             END;
;  646          0640  2
;  647          0641  2         TES;
;  648          0642  2
;  649          0643  2     !
;  650          0644  2     ! Step through the Service Data Table to find all parameters in the requested
;  651          0645  2     ! database.  Move these parameter's field IDs into the NFB so that NETACP
;  652          0646  2     ! will return their values in the P4 buffer.
;  653          0647  2     !
;  654          0648  2     INCR svd_index FROM 0 TO svd$c_entry_count DO
;  655          0649  3         BEGIN
;  656          0650  3         IF .mom$ab_service_data [.svd_index, svd$b_nfb_database]
;  657          0651  3                                         EQL .database THEN
;  658          0652  4             BEGIN
;  659          0653  4             nfb [nfb$l_fldid] = .mom$ab_service_data [.svd_index, svd$l_nfb_id];
;  660          0654  4             nfb = .nfb + 4;
;  661          0655  3             END;
;  662          0656  2         END;
;  663          0657  2     nfb [nfb$l_fldid] = 0;
;  664          0658  2
;  665          0659  2     nfbdsc [0] = nfb [nfb$l_fldid] + 4 - nfb_buffer;
;  666          0660  2     nfbdsc [1] = nfb_buffer;
;  667          0661  2     !
;  668          0662  2     ! If there is an entry in the volatile data base then NETACP will return the
;  669          0663  2     ! data requested in the NFB.  Return this data to the calling routine.
;  670          0664  2     !
;  671          0665  2     STATUS = mom$netacp_qio (nfbdsc,
;  672          0666  2                             p2_dsc,
;  673          0667  2                             p4_buf_dsc [0],
;  674          0668  2                             .p4_buf_dsc);
;  675          0669  2
;  676          0670  2     IF NOT .status THEN
;  677          0671  3         BEGIN
;  678          0672  3         mom$bld_reply (mom$ab_msgblock, msgsize);
;  679          0673  3         $signal_msg (mom$ab_nice_xmit_buf, .msgsize);
;  680          0674  2         END;
;  681          0675  2
;  682          0676  2     RETURN .status;
;  683          0677  1     END;                            ! of mom$get_voldb_data



                                                   007C 00000          .ENTRY  MOM$GET_VOLDB_DATA, Save R2,R3,R4,R5,R6      ; 0517
                                   56 00000000'  00  9E 00002          MOVAB   MOM$Q_P2_BUF_DSC, R6                          ;
                                   5E      FDE0  CE  9E 00009          MOVAB   -544(SP), SP                                  ;
    0200    8F             00      6E            00  2C 0000E          MOVC5   #0, (SP), #0, #512, NFB_BUFFER               ; 0562
```

```
                                  10    AE     00015                                                      ;
                      52          10    AE  9E 00017           MOVAB   NFB_BUFFER, NFB                    ; 0563
                      62                22  90 0001B           MOVB    #34, (NFB)                         ; 0564
                      54          04    AC  D0 0001E           MOVL    DATABASE, R4                       ; 0565
                02    A2                54  9B 00022           MOVZBW  R4, 2(NFB)                         ;
                08    A2                01  D0 00026           MOVL    #1, 8(NFB)                         ; 0567
                                  0C    A2  94 0002A           CLRB    12(NFB)                            ; 0568
                      02                54  D1 0002D           CMPL    R4, #2                             ; 0576
                                        79  12 00030           BNEQ    4$                                 ;
                      50 00000000G      00  9A 00032           MOVZBL  MOM$GB_ENTITY_CODE, R0             ; 0577
                      02                50  91 00039           CMPB    R0, #2                             ; 0580
                                        29  12 0003C           BNEQ    1$                                 ;
                04    A2 02020044       8F  D0 0003E           MOVL    #33685572, 4(NFB)                  ; 0582
                                        5E  DD 00046           PUSHL   SP                                 ; 0590
                                  08    AE  9F 00048           PUSHAB  LENGTH                             ;
                                  10    AE  9F 0004B           PUSHAB  KEY                                ;
           00000000V  00                03  FB 0004E           CALLS   #3, MOM_GET_CIRC_SEARCH2_KEY       ;
                08    A2          08    AE  D0 00055           MOVL    KEY, 8(NFB)                        ; 0591
                                  F8    AD  9F 0005A           PUSHAB  P2_DSC                             ; 0592
                                        56  DD 0005D           PUSHL   R6                                 ;
                                  08    AE  DD 0005F           PUSHL   ADDRESS                            ; 0594
                                  10    AE  DD 00062           PUSHL   LENGTH                             ;
                                        3B  11 00065           BRB     3$                                 ; 0593
                                        50  D5 00067 1$:       TSTL    R0                                 ; 0598
                                        20  12 00069           BNEQ    2$                                 ;
                04    A2 02010012       8F  D0 0006B           MOVL    #33619986, 4(NFB)                  ; 0600
                                  F8    AD  9F 00073           PUSHAB  P2_DSC                             ; 0601
                                        56  DD 00076           PUSHL   R6                                 ;
                                        7E  D4 00078           CLRL    -(SP)                              ;
                      7E                01  CE 0007A           MNEGL   #1, -(SP)                          ; 0603
                      50 00000000G      00  D0 0007D           MOVL    MOM$GQ_ENTITY_BUF_DSC+4, R0        ; 0602
                      7E                60  3C 00084           MOVZWL  (R0), -(SP)                        ;
                                        7E  D4 00087           CLRL    -(SP)                              ; 0601
                                        61  11 00089           BRB     7$                                 ;
                      01                50  91 0008B 2$:       CMPB    R0, #1                             ; 0607
                                        63  12 0008E           BNEQ    8$                                 ;
                04    A2 02020043       8F  D0 00090           MOVL    #33685571, 4(NFB)                  ; 0609
                                  F8    AD  9F 00098           PUSHAB  P2_DSC                             ; 0610
                                        56  DD 0009B           PUSHL   R6                                 ;
                                        7E  D4 0009D           CLRL    -(SP)                              ;
                      7E                01  CE 0009F           MNEGL   #1, -(SP)                          ; 0612
                      7E 00000000G      00  7D 000A2 3$:       MOVQ    MOM$GQ_ENTITY_BUF_DSC, -(SP)       ; 0610
                                        41  11 000A9           BRB     7$                                 ;
                      05                54  D1 000AB 4$:       CMPL    R4, #5                             ; 0617
                                        43  12 000AE           BNEQ    8$                                 ;
                04    A2 05020041       8F  D0 000B0           MOVL    #84017217, 4(NFB)                  ; 0619
                      53 00000000*      00  9A 000B8           MOVZBL  <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_SLI*137>-; 0627
                                                                       >+8>, LINE_LEN                     ;
   00000000*    00    53                2E  3A 000BF           LOCC    #46, LINE_LEN, <<MOM$AB_SERVICE_DATA+-  ; 0629
                                                                       <SVD$GK_PCNO_SLI*137>>+9>          ;
                                        02  12 000C7           BNEQ    5$                                 ;
                                        51  D4 000C9           CLRL    R1                                 ;
                                        51  D5 000CB 5$:       TSTL    PERIOD_PTR                         ; 0631
                                        0B  13 000CD           BEQL    6$                                 ;
                      50 00000000*      00  9E 000CF           MOVAB   <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_SLI*137>-; 0633
                                                                       >+9>, R0                           ;
         53           51                50  C3 000D6           SUBL3   R0, PERIOD_PTR, LINE_LEN           ;
```

```
                                F8    AD   9F 000DA 6$:      PUSHAB  P2_DSC                                          ; 0636
                                      56   DD 000DD          PUSHL   R6                                              ;
                                      7E   D4 000DF          CLRL    -(SP)                                           ;
                         7E           01   CE 000E1          MNEGL   #1, -(SP)                                       ; 0637
                     00000000*        00   9F 000E4          PUSHAB  <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_SLI*137>-; 0636
                                                                     >+9>                                            ;
                                      53   DD 000EA          PUSHL   LINE_LEN                                        ;
              00000000G  00           06   FB 000EC 7$:      CALLS   #6, MOM$BUILD_P2                                ;
                         50           01   CE 000F3 8$:      MNEGL   #1, SVD_INDEX                                   ; 0648
                                      22   11 000F6          BRB     10$                                             ;
                 51      50 00000089  8F   C5 000F8 9$:      MULL3   #137, SVD_INDEX, R1                             ; 0650
         54 00000000G0041         08  00   ED 00100          CMPZV   #0, #8, MOM$AB_SERVICE_DATA+3[R1], R4           ; 0651
                                      0E   12 0010A          BNEQ    10$                                             ;
                     00000000G0041    9F      0010C          PUSHAB  MOM$AB_SERVICE_DATA[R1]                         ; 0653
                         10  A2       9E   D0 00113          MOVL    @(SP)+, 16(NFB)                                 ;
                             52       04   C0 00117          ADDL2   #4, NFB                                         ; 0654
                 D6          50 00000000G  8F F3 0011A 10$:  AOBLEQ  #SVD$C_ENTRY_COUNT, SVD_INDEX, 9$               ; 0648
                                10    A2   D4 00122          CLRL    16(NFB)                                         ; 0657
                         50     10    AE   9E 00125          MOVAB   NFB_BUFFER, R0                                  ; 0659
                         52           50   C2 00129          SUBL2   R0, R2                                          ;
                   F0    AD     14    A2   9E 0012C          MOVAB   20(R2), NFBDSC                                  ;
                   F4    AD     10    AE   9E 00131          MOVAB   NFB_BUFFER, NFBDSC+4                            ; 0660
                                08    AC   DD 00136          PUSHL   P4_BUF_DSC                                      ; 0668
                                08    AC   DD 00139          PUSHL   P4_BUF_DSC                                      ; 0667
                                F8    AD   9F 0013C          PUSHAB  P2_DSC                                          ; 0665
                                F0    AD   9F 0013F          PUSHAB  NFBDSC                                          ;
              00000000G  00           04   FB 00142          CALLS   #4, MOM$NETACP_QIO                              ; 0667
                         52           50   D0 00149          MOVL    R0, STATUS                                      ;
                         26           52   E8 0014C          BLBS    STATUS, 11$                                     ; 0670
                                0C    AE   9F 0014F          PUSHAB  MSGSIZE                                         ; 0672
                     00000000G        00   9F 00152          PUSHAB  MOM$AB_MSGBLOCK                                 ;
              00000000G  00           02   FB 00158          CALLS   #2, MOM$BLD_REPLY                               ;
                                0C    AE   DD 0015F          PUSHL   MSGSIZE                                         ; 0673
                     00000000G        00   9F 00162          PUSHAB  MOM$AB_NICE_XMIT_BUF                            ;
                     02070000         8F   DD 00168          PUSHL   #34013184                                       ;
              00000000G  00           03   FB 0016E          CALLS   #3, LIB$SIGNAL                                  ;
                         50           52   D0 00175 11$:     MOVL    STATUS, R0                                      ; 0676
                                           04 00178          RET                                                     ; 0677

; Routine Size:  377 bytes,    Routine Base:  $CODE$ + 0304
```

```
1 %SBTTL 'mom_get_circ_search2_key'
1 GLOBAL ROUTINE mom_get_circ_search2_key (key, length, address) : NOVALUE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !       This routine is called when preparing to get service data for
1 !       the target from the volatile database.  At this point the entity
1 !       is always MOM$C_CIRCUIT, and the operation is a TRIGGER VIA, a
1 !       LOAD VIA, or autoservice.  In these three cases, there is no node
1 !       ID with which to locate the target in the node volatile database.
1 !       For point to point circuits, it is sufficient to look for a node
1 !       with a service circuit matching the one from the command.  For
1 !       NI circuits, this routine sets up the second search key to match
1 !       in the database.
1 !
1 ! FORMAL PARAMETERS:
1 !       KEY     Address to return search key two ID
1 !       LENGTH  Address to return search key two length
1 !       ADDRESS Address to return search key two address.
1 !
1 !--
1
2 BEGIN
2
2 LOCAL
2     physical_addr_ptr;
2
2 !
2 ! At this point the NICE message (operservice) or initial MOP message (auto-
2 ! service) has been parsed, and the only parameters in the Service Data table
2 ! are from this message.  Therefore, the presence of the NI physical address
2 ! in the SVD is an indication that the service circuit is an NI.
2 !
2 IF .mom$ab_service_data [svd$gk_pcno_pha, svd$v_msg_param] THEN
2     !
2     ! If the Physical Address begins with the DEC assigned NI prefix, then
2     ! the last word of the Physical Address is the target node's address.
2     ! Extract it an use it as the second search key to find the target in
2     ! the volatile database (it would actually be sufficient by itself).
2     !
3     BEGIN
3     physical_addr_ptr = mom$ab_service_data [svd$gk_pcno_pha, svd$t_string];
3     IF ..physical_addr_ptr EQL mom$k_ni_prefix THEN
4         BEGIN
4         .key = nfb$c_ndi_add;
4         .length = 0;
4         .address = .(.physical_addr_ptr + 4)<0,16>;
4         END
3     ELSE
3         !
3         ! Build a P2 buffer that uses the NI hardware address (the entire
3         ! physical address) to find the target's entry in NETACP's node
3         ! database.
3         !
4         BEGIN
4         .key = nfb$c_ndi_hwa;
4         .length = 6;
```

```
;  742          0735 4            .address = .physical_addr_ptr;
;  743          0736 3            END;
;  744          0737 3         END
;  745          0738 2  ELSE
;  746          0739 2         !
;  747          0740 2         ! The circuit is point-to-point or multipoint.  The service circuit
;  748          0741 2         ! IDs in the node volatile database must be unique for these.
;  749          0742 2         !
;  750          0743 3         BEGIN
;  751          0744 3         .key = nfb$c_wildcard;
;  752          0745 3         .length = -1;
;  753          0746 3         .address = 0;
;  754          0747 2         END;
;  755          0748 1  END;                                    ! End of mom_get_circ_search2_key



                                              0000 00000           .ENTRY  MOM_GET_CIRC_SEARCH2_KEY, Save nothing          ; 0679
                        32 00000000*  00   E9 00002            BLBC    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_PHA*137>-;  0711
                                                                       >+7>, 2$
                        50 00000000*  00   9E 00009            MOVAB   <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_PHA*137>-;  0719
                                                                       >+9>, PHYSICAL_ADDR_PTR
           000400AA  8F               60   D1 00010            CMPL    (PHYSICAL_ADDR_PTR), #262314                    ; 0720
                                      11   12 00017            BNEQ    1$                                              ;
                 04  BC 02010012      8F   D0 00019            MOVL    #33619986, @KEY                                 ; 0722
                                  08  BC   D4 00021            CLRL    @LENGTH                                         ; 0723
                 0C  BC           04  A0   3C 00024            MOVZWL  4(PHYSICAL_ADDR_PTR), @ADDRESS                  ; 0724
                                           04 00029            RET                                                     ; 0720
                 04  BC 02020057      8F   D0 0002A 1$:        MOVL    #33685591, @KEY                                 ; 0733
                 08  BC                06   D0 00032            MOVL    #6, @LENGTH                                     ; 0734
                 0C  BC                50   D0 00036            MOVL    PHYSICAL_ADDR_PTR, @ADDRESS                     ; 0735
                                           04 0003A            RET                                                     ; 0711
                 04  BC                01   D0 0003B 2$:        MOVL    #1, @KEY                                        ; 0744
                 08  BC                01   CE 0003F            MNEGL   #1, @LENGTH                                     ; 0745
                                  0C  BC   D4 00043            CLRL    @ADDRESS                                        ; 0746
                                           04 00046            RET                                                     ; 0748

; Routine Size:  71 bytes,    Routine Base:  $CODE$ + 047D
;  757          0749  1 %SBTTL 'mom$bldmoprds  Build MOP mode running message'
;  758          0750  1 GLOBAL ROUTINE mom$bldmoprds (msgdsc) : NOVALUE =
;  759          0751  1
;  760          0752  1 !++
;  761          0753  1 ! FUNCTIONAL DESCRIPTION:
;  762          0754  1 !
;  763          0755  1 !       This routine builds a 'MOP Request Dump Service' message in the
;  764          0756  1 !       MOP transmit buffer.
;  765          0757  1 !
;  766          0758  1 ! FORMAL PARAMETERS:
;  767          0759  1 !
;  768          0760  1 !       MSGDSC
;  769          0761  1 !
;  770          0762  1 !--
;  771          0763  1
;  772          0764  2 BEGIN
;  773          0765  2
;  774          0766  2 MAP
;  775          0767  2     msgdsc : REF VECTOR;
;  776          0768  2 !
;  777          0769  2 ! Move the 'MOP request dump service' function code into the buffer.
;  778          0770  2 !
;  779          0771  2 CH$WCHAR (mop$_fct_rds, mom$ab_mop_xmit_buf);
;  780          0772  2 !
;  781          0773  2 ! Set up the descriptor for the return.
;  782          0774  2 !
;  783          0775  2 msgdsc [0] = 1;
;  784          0776  2 msgdsc [1] = mom$ab_mop_xmit_buf;
;  785          0777  2
;  786          0778  1 END;                                  ! End of MOM$BLDMOPRDS



                                        0004 00000          .ENTRY  MOM$BLDMOPRDS, Save R2                          ; 0750
                     52 00000000G  00    9E 00002          MOVAB   MOM$AB_MOP_XMIT_BUF, R2                         ;
                     62            0C    90 00009          MOVB    #12, MOM$AB_MOP_XMIT_BUF                        ; 0771
                     50        04  AC    D0 0000C          MOVL    MSGDSC, R0                                      ; 0775
                     60            01    D0 00010          MOVL    #1, (R0)                                        ;
                  04 A0            62    9E 00013          MOVAB   MOM$AB_MOP_XMIT_BUF, 4(R0)                      ; 0776
                                         04 00017          RET                                                     ; 0778

; Routine Size:  24 bytes,    Routine Base:  $CODE$ + 04C4
```

```
1 %SBTTL 'mom$bldmopboot  Build enter MOP mode message'
1 GLOBAL ROUTINE mom$bldmopboot (msgdsc) : NOVALUE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine builds the 'Boot' (trigger) message in the
1 !       MOP transmit buffer.  This is the old 'Enter MOP Mode' message.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       MSGDSC
1 !
1 !--
1
2 BEGIN
2
2 MAP
2     msgdsc : REF VECTOR;
2
2 LOCAL
2     db_passwd_len,
2     msg_passwd_len,
2     ptr,
2     status;
2
2 !
2 ! Build the 'Boot' message.
2 !
2 ptr = mom$ab_mop_xmit_buf;
2 CH$WCHAR_A (mop$_fct_emm, ptr);
2 !
2 ! Move the service password from the Service Data base to the MOP message.
2 ! If no password is set then zeros will be used.  The MOP trigger password
2 ! is always four bytes for point to point and 8 bytes for NI.
2 !
2 db_passwd_len = .mom$ab_service_data [svd$gk_pcno_spa, svd$b_string_len];
2 msg_passwd_len = .db_passwd_len;
2 IF .mom$gl_service_flags [mom$v_ni_circ] THEN
2     msg_passwd_len = 8
2 ELSE
2     msg_passwd_len = 4;
2 ptr = CH$COPY (.db_passwd_len,
2               mom$ab_service_data [svd$gk_pcno_spa, svd$t_string],
2               0, .msg_passwd_len, .ptr);
2 !
2 ! The MOP V2.1 Boot message has an 8 byte password (the old version has a
2 ! 4 byte one) and some extra fields.  Add those extra fields.
2 !
2 IF .msg_passwd_len GTR 4 THEN
3     BEGIN
3     IF .mom$gl_service_flags [mom$v_console_carrier_load] THEN
3         CH$WCHAR_A (mop$c_pro_com, ptr)         ! Load communications processor
3     ELSE
3         CH$WCHAR_A (mop$c_pro_sys, ptr);        ! Load system processor
3     IF .mom$gb_function EQL nma$c_fnc_tri THEN
3         !
```

```
;  845       0836  3              ! Control: Boot server = system default,
;  846       0837  3              !          Boot device = system default
;  847       0838  3              CH$WCHAR_A (0, ptr)
;  848       0839  3          ELSE
;  849       0840  3              !
;  850       0841  3              ! For load triggers, tell the target to request the load from this
;  851       0842  3              ! system (as opposed to multicasting the load request).
;  852       0843  3              ! Control: Boot server = requesting system,
;  853       0844  3              !          Boot device = system default
;  854       0845  3              CH$WCHAR_A (1, ptr);
;  855       0846  3          !
;  856       0847  3          !
;  857       0848  3          ! Software ID - always boot for operating system.  I don't see any way
;  858       0849  3          ! for me to tell if I'm loading diagnostics or not.
;  859       0850  3          !
;  860       0851  3          CH$WCHAR_A (-1, ptr);
;  861       0852  2          END;
;  862       0853  2  !
;  863       0854  2  ! Set up the descriptor for the return.
;  864       0855  2  !
;  865       0856  2  msgdsc [0] = .ptr - mom$ab_mop_xmit_buf;
;  866       0857  2  msgdsc [1] = mom$ab_mop_xmit_buf;
;  867       0858  2
;  868       0859  1  END;                                  ! End of mom$bldmopboot



                                       01FC 00000          .ENTRY  MOM$BLDMOPBOOT, Save R2,R3,R4,R5,R6,R7,R8     ; 0780
                    58 00000000G    00   9E 00002          MOVAB   MOM$GL_SERVICE_FLAGS, R8                      ;
                    57 00000000G    00   9E 00009          MOVAB   MOM$AB_MOP_XMIT_BUF, R7                       ;
                    53              67   9E 00010          MOVAB   MOM$AB_MOP_XMIT_BUF, PTR                      ; 0808
                    83              06   90 00013          MOVB    #6, (PTR)+                                    ; 0809
                    50 00000000*    00   9A 00016          MOVZBL  <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_SPA*137>-; 0815
                                                                   >+8>, DB_PASSWD_LEN
                    56              50   D0 0001D          MOVL    DB_PASSWD_LEN, MSG_PASSWD_LEN                 ; 0816
               05   68              01   E1 00020          BBC     #1, MOM$GL_SERVICE_FLAGS, 1$                  ; 0817
                    56              08   D0 00024          MOVL    #8, MSG_PASSWD_LEN                            ; 0818
                    03              11      00027          BRB     2$                                            ;
                    56              04   D0 00029 1$:      MOVL    #4, MSG_PASSWD_LEN                            ; 0820
      56       00 00000000*    00   50   2C 0002C 2$:      MOVC5   DB_PASSWD_LEN, <<MOM$AB_SERVICE_DATA+-        ; 0823
                                    63      00035                  <SVD$GK_PCNO_SPA*137>>+9>, #0, -              ;
                                                                   MSG_PASSWD_LEN, (PTR)                         ;
                    04              56   D1 00036          CMPL    MSG_PASSWD_LEN, #4                            ; 0828
                    22              15      00039          BLEQ    7$                                            ;
               05   68              06   E1 0003B          BBC     #6, MOM$GL_SERVICE_FLAGS, 3$                  ; 0830
                    63              01   90 0003F          MOVB    #1, (PTR)                                     ; 0831
                    02              11      00042          BRB     4$                                            ; 0833
                    63              94      00044 3$:      CLRB    (PTR)                                         ;
                    53              D6      00046 4$:      INCL    PTR                                           ; 0831
                    11 00000000G    00   91 00048          CMPB    MOM$GB_FUNCTION, #17                          ; 0834
                    04              12      0004F          BNEQ    5$                                            ;
                    63              94      00051          CLRB    (PTR)                                         ; 0838
                    03              11      00053          BRB     6$                                            ; 0845
                    63              01   90 00055 5$:      MOVB    #1, (PTR)                                     ;
                    53              D6      00058 6$:      INCL    PTR                                           ; 0838
```

```
                          83          01  8E 0005A         MNEGB   #1, (PTR)+                                     ; 0851
                          50      04  AC  D0 0005D 7$:     MOVL    MSGDSC, R0                                     ; 0856
                          51          67  9E 00061         MOVAB   MOM$AB_MOP_XMIT_BUF, R1                        ;
               60         53          51  C3 00064         SUBL3   R1, PTR, (R0)                                  ;
                     04   A0          67  9E 00068         MOVAB   MOM$AB_MOP_XMIT_BUF, 4(R0)                     ; 0857
                                          04 0006C         RET                                                    ; 0859

; Routine Size:  109 bytes,    Routine Base:  $CODE$ + 04DC
```

```
1 %SBTTL 'mom$bldmopplt   Build MOP Parameter Load with Transfer Address message'
1 GLOBAL ROUTINE mom$bldmopplt (plt_msg_dsc, load_seg_num,
1                                          transfer_addr) : NOVALUE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !       This routine is called to build the MOP Parameter Load with
1 !       Transfer message which is sent to the target node at the end of
1 !       a down line load.
1 !
1 ! FORMAL PARAMETERS:
1 !       PLT_MSG_DSC - Descriptor of buffer for MOP Parameter Load with
1 !               Transfer message.
1 !       LOAD_SEG_NUM - Number of load segments loaded modulo 256.
1 !       TRANSFER_ADDR - Address to start executing image just loaded.
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !--
1
2 BEGIN
2
2 MAP
2     plt_msg_dsc : REF VECTOR,
2     load_seg_num: BYTE;
2
2 LOCAL
2     len,
2     ptr,
2     node_addr: WORD,
2     date_time : VECTOR [7, WORD],
2     century,
2     year;
2
2 !
2 ! If the load file was a bootstrap then send an empty memory load with
2 ! transfer address message.
2 !
2 IF .mom$ab_service_data [svd$gk_pcno_sty, svd$l_param] NEQU nma$c_soft_osys THEN
3     BEGIN
3
3     ptr = mom$ab_mop_xmit_buf;
3
3     CH$WCHAR_A (mop$_fct_mlt, ptr);                 ! Function code
3     CH$WCHAR_A (.load_seg_num, ptr);                ! Load segment number
3     (.PTR)<0,32> = 0;                               ! Zero load address
3     ptr = .ptr + 4;                                 ! Skip load address
3 !
3 ! Output the MOP message to the debug log.
3 !
3     mom$debug_txt (dbg$c_srvtrc,
4             $ASCID ('Transmitting empty memory load with transfer address.')
3             );
3
3     END
```

```
2 ELSE
3     BEGIN
3 !
3 ! The load file was the system image so send a parameter load with transfer
3 ! address message.
3 !
3     ptr = mom$ab_mop_xmit_buf;
3     CH$WCHAR_A (mop$_fct_plt, ptr);                    ! Function code
3     CH$WCHAR_A (.load_seg_num, ptr);                   ! Load segment number
3 !
3 ! If target node name specified then add it to message.
3 !
3     len = .mom$ab_service_data [svd$gk_pcno_nna, svd$b_string_len];
3     IF .len NEQ 0 THEN
4         BEGIN
4         CH$WCHAR_A (mop$c_par_nna, ptr);               ! Parameter code
4         CH$WCHAR_A (.len, ptr);                        ! Name length
4         PTR = CH$MOVE (.len,                           ! Name
4                         mom$ab_service_data [svd$gk_pcno_nna, svd$t_string],
4                         .ptr);
3         END;
3 !
3 ! Add target node address to message.  If address not specified then
3 ! program error.
3 !
3     CH$WCHAR_A (mop$c_par_nad, ptr);     ! Parameter code
3     CH$WCHAR_A (2, ptr);                 ! Address length
3     node_addr = .mom$ab_service_data [svd$gk_pcno_add, svd$l_param];
3     !
3     ! If it's a phase III node, mask out the area number in the node address.
3     ! DECnet Phase III did not include areas.
3     !
3     IF .mom$ab_service_data [svd$gk_pcno_snv, svd$l_param] EQL nma$c_nodsnv_ph3
3     THEN
4         BEGIN
4         MAP node_addr: BBLOCK;
4         node_addr [nma$v_area] = 0;
3         END;
3     ptr = CH$MOVE (2, node_addr, .ptr);
3
3 !
3 ! If the host node name is specified then add it to the message.
3 !
3     len = .mom$ab_service_data [svd$gk_pcno_$hna, svd$b_string_len];
3     IF .len NEQ 0 THEN
4         BEGIN
4         CH$WCHAR_A (mop$c_par_hna, ptr);               ! Parameter code
4         CH$WCHAR_A (.len,                              ! Name length
4                     ptr);
4         PTR = CH$MOVE (.len,                           ! Name
4                        mom$ab_service_data [svd$gk_pcno_$hna, svd$t_string],
4                        .ptr);
3         END;
3 !
3 ! If the host address is specified then add it to the message.
3 !
3     IF .mom$ab_service_data [svd$gk_pcno_iho, svd$l_param] NEQ 0 THEN
```

```
4             BEGIN
4             CH$WCHAR_A (mop$c_par_had, ptr);          ! Parameter code
4             CH$WCHAR_A (2, ptr);                   ! Address length
4             node_addr = .mom$ab_service_data [svd$gk_pcno_iho, svd$l_param];
4             !
4             ! If it's a phase III node, mask out the area number in the node address.
4             !
4             IF .mom$ab_service_data [svd$gk_pcno_snv, svd$l_param] EQL
4                                                 nma$c_nodsnv_ph3 THEN
5                 BEGIN
5                 MAP node_addr: BBLOCK;
5                 node_addr [nma$v_area] = 0;
4                 END;
4             ptr = CH$MOVE (2, node_addr, .ptr);
3             END;
3
3         !
3         ! If it's not a phase III node, add the system time to the message
3         !
3         IF .mom$ab_service_data [svd$gk_pcno_snv, svd$l_param] NEQ
3                                                 nma$c_nodsnv_ph3 THEN
4             BEGIN
4             CH$WCHAR_A (mop$c_par_hti, ptr);
4             CH$WCHAR_A (10, ptr);
4             $NUMTIM (TIMBUF = date_time);
4             !
4             ! The parameter load with transfer message requires that the year be
4             ! broken up into a century and a year.  Do that.
4             !
4             century = .date_time [0] /100;
4             year = .date_time [0] MOD 100;
4             !
4             ! The rest of the date/time string required in the MOP Parameter Load with
4             ! Transfer message is in the same order as that returned by the $NUMTIM
4             ! system service.  Put the string into the MOP message, converting the words
4             ! to bytes.
4             !
4             CH$WCHAR_A (.century, ptr);
4             CH$WCHAR_A (.year, ptr);
4             INCR i FROM 1 TO 6 DO
4                 CH$WCHAR_A (.date_time [.i], ptr);
4             !
4             ! Fill in the Time Differential Factor hours and minutes as 0.  VMS
4             ! doesn't keep Greenwich Mean Time around for figuring these out with.
4             !
4             ptr = CH$FILL (0, 2, .ptr);
3             END;
3
3         !
3         ! Add the end mark.
3         !
3         CH$WCHAR_A (0, ptr);
3         !
3         ! Output the trace message.
3         !
3         mom$debug_txt ( dbg$c_srvtrc,
4                 $ASCID ('Transmitting parameter load with transfer address.')
; 1041      1031  3               );
; 1042      1032  3
; 1043      1033  2         END;
; 1044      1034  2     !
; 1045      1035  2     ! Add transfer address.
; 1046      1036  2     !
; 1047      1037  2     ptr = CH$MOVE (4, transfer_addr, .ptr);
; 1048      1038  2     !
; 1049      1039  2     ! Send the message.
; 1050      1040  2     !
; 1051      1041  2     plt_msg_dsc [0]  = .ptr - mom$ab_mop_xmit_buf;
; 1052      1042  2     plt_msg_dsc [1] = mom$ab_mop_xmit_buf;
; 1053      1043  1     END;                              ! End of MOM$BLDMOPPLT


                                                                      .PSECT  $PLIT$,NOWRT,NOEXE,2

6D 65 20 67 6E 69 74 74 69 6D 73 6E 61 72 54 00018 P.AAE:  .ASCII  \Transmitting empty memory load with tran\    ;
64 61 6F 6C 20 79 72 6F 6D 65 6D 20 79 74 70 00027                                                          ;
               6E 61 72 74 20 68 74 69 77 20 00036                                                           ;
      2E 73 73 65 72 64 64 61 20 72 65 66 73 00040         .ASCII  \sfer address.\                          ;
                                             0004D         .BLKB   3
                                   00000035  00050 P.AAD:  .LONG   53                                       ;
                                   00000000' 00054         .ADDRESS P.AAE
61 70 20 67 6E 69 74 74 69 6D 73 6E 61 72 54 00058 P.AAG:  .ASCII  \Transmitting parameter load with transfe\ ;
69 77 20 64 61 6F 6C 20 72 65 74 65 6D 61 72 00067                                                          ;
               65 66 73 6E 61 72 74 20 68 74 00076                                                          ;
               2E 73 73 65 72 64 64 61 20 72 00080         .ASCII  \r address.\                             ;
                                             0008A         .BLKB   2
                                   00000032  0008C P.AAF:  .LONG   50                                       ;
                                   00000000' 00090         .ADDRESS P.AAG                                   ;

                                                           .EXTRN  SYS$NUMTIM

                                                           .PSECT  $CODE$,NOWRT,2

                                        07FC 00000         .ENTRY  MOM$BLDMOPPLT, Save R2,R3,R4,R5,R6,R7,R8,-    ; 0861
                                                                   R9,R10                                       ;
                      5A 00000000G  00   9E 00002         MOVAB   MOM$AB_MOP_XMIT_BUF, R10                       ;
                      5E            10   C2 00009         SUBL2   #16, SP                                        ;
                      53            6A   9E 0000C         MOVAB   MOM$AB_MOP_XMIT_BUF, PTR                       ; 0903
                      02 00000000*  00   D1 0000F         CMPL    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_STY*137>-; 0900
                                                                  >+9>, #2                                       ;
                                    11   13 00016         BEQL    1$                                             ;
                                    83   94 00018         CLRB    (PTR)+                                         ; 0905
                      83       08   AC   90 0001A         MOVB    LOAD_SEG_NUM, (PTR)+                           ; 0906
                                    83   D4 0001E         CLRL    (PTR)+                                         ; 0907
                         00000000'  00   9F 00020         PUSHAB  P.AAD                                          ; 0913
                                  00BF   31 00026         BRW     9$                                             ; 0912
                      83            14   90 00029 1$:     MOVB    #20, (PTR)+                                    ; 0924
                      83       08   AC   90 0002C         MOVB    LOAD_SEG_NUM, (PTR)+                           ; 0925
                      56 00000000*  00   9A 00030         MOVZBL  <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_NNA*137>-; 0929
                                                                  >+8>, LEN                                      ;
                                    0E   13 00037         BEQL    2$                                             ; 0930
                      83            01   90 00039         MOVB    #1, (PTR)+                                     ; 0932
                      83            56   90 0003C         MOVB    LEN, (PTR)+                                    ; 0933
```

```
                        63 00000000*  00              56  28 0003F          MOVC3    LEN, <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_NNA-;  0936
                                                                                     *137>>+9>, (PTR)                                :
                                      83      0202    8F  B0 00047 2$:      MOVW     #514, (PTR)+                                    :  0942
                                      57 00000000*    00  B0 0004C          MOVW     <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_ADD*137>-;  0944
                                                                                     >+9>, NODE_ADDR                                 :
                                      58 00000000*    00  D0 00053          MOVL     <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_SNV*137>-;  0949
                                                                                     >+9>, R8                                        :
                                                      59  D4 0005A          CLRL     R9                                              :
                                                      58  D5 0005C          TSTL     R8                                              :
                                                      07  12 0005E          BNEQ     3$                                              :
                                                      59  D6 00060          INCL     R9                                              :
                                      57      FC00    8F  AA 00062          BICW2    #64512, NODE_ADDR                               :  0953
                                      83              57  B0 00067 3$:      MOVW     NODE_ADDR, (PTR)+                               :  0955
                                      56 00000000*    00  9A 0006A          MOVZBL   <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_$HNA*-      :  0960
                                                                                     137>>+8>, LEN                                   :
                                                      0E  13 00071          BEQL     4$                                              :  0961
                                      83              03  90 00073          MOVB     #3, (PTR)+                                      :  0963
                                      83              56  90 00076          MOVB     LEN, (PTR)+                                     :  0965
                        63 00000000*  00              56  28 00079          MOVC3    LEN, <<MOM$AB_SERVICE_DATA+-                    :  0968
                                                                                     <SVD$GK_PCNO_$HNA*137>>+9>, (PTR)               :
                                      50 00000000*    00  D0 00081 4$:      MOVL     <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_IHO*137>-;  0973
                                                                                     >+9>, R0                                        :
                                                      13  13 00088          BEQL     6$                                              :
                                      83      0204    8F  B0 0008A          MOVW     #516, (PTR)+                                    :  0975
                                      57              50  B0 0008F          MOVW     R0, NODE_ADDR                                   :  0977
                                      05              59  E9 00092          BLBC     R9, 5$                                          :  0981
                                      57      FC00    8F  AA 00095          BICW2    #64512, NODE_ADDR                               :  0985
                                      83              57  B0 0009A 5$:      MOVW     NODE_ADDR, (PTR)+                               :  0987
                                                      58  D5 0009D 6$:      TSTL     R8                                              :  0993
                                                      3F  13 0009F          BEQL     8$                                              :
                                      83      0A05    8F  B0 000A1          MOVW     #2565, (PTR)+                                   :  0996
                                                      7E  D4 000A6          CLRL     -(SP)                                           :  0998
                                                04    AE  9F 000A8          PUSHAB   DATE_TIME                                       :
                           00000000G  00              02  FB 000AB          CALLS    #2, SYS$NUMTIM                                  :
                                      51              6E  3C 000B2          MOVZWL   DATE_TIME, CENTURY                              :  1003
                                      51 00000064     8F  C6 000B5          DIVL2    #100, CENTURY                                   :
                                      50              6E  3C 000BC          MOVZWL   DATE_TIME, YEAR                                 :  1004
         7E             00            50              01  7A 000BF          EMUL     #1, YEAR, #0, -(SP)                             :
         50             50            8E 00000064     8F  7B 000C4          EDIV     #100, (SP)+, YEAR, YEAR                         :
                                      83              51  90 000CD          MOVB     CENTURY, (PTR)+                                 :  1011
                                      83              50  90 000D0          MOVB     YEAR, (PTR)+                                    :  1012
                                      50              01  D0 000D3          MOVL     #1, I                                           :  1014
                                      83            6E40  33 000D6 7$:      CVTWB    DATE_TIME[I], (PTR)+                            :
                        F8            50              06  F3 000DA          AOBLEQ   #6, I, 7$                                       :
                                                      83  B4 000DE          CLRW     (PTR)+                                          :  1019
                                                      83  94 000E0 8$:      CLRB     (PTR)+                                          :  1025
                                         00000000'    00  9F 000E2          PUSHAB   P.AAF                                           :  1030
                                                      06  DD 000E8 9$:      PUSHL    #6                                              :  1029
                           00000000G  00              02  FB 000EA          CALLS    #2, MOM$DEBUG_TXT                               :
                                      83        0C    AC  D0 000F1          MOVL     TRANSFER_ADDR, (PTR)+                           :  1037
                                      50        04    AC  D0 000F5          MOVL     PLT_MSG_DSC, R0                                 :  1041
                                      51              6A  9E 000F9          MOVAB    MOM$AB_MOP_XMIT_BUF, R1                         :
                        60            53              51  C3 000FC          SUBL3    R1, PTR, (R0)                                   :
                                04    A0              6A  9E 00100          MOVAB    MOM$AB_MOP_XMIT_BUF, 4(R0)                      :  1042
                                                          04 00104          RET                                                      :  1043

; Routine Size:  261 bytes,    Routine Base:  $CODE$ + 0549
; 1054          1044  1
; 1055          1045  1 END
; 1056          1046  1
; 1057          1047  0 ELUDOM



                                                                            .EXTRN  LIB$SIGNAL

;                                     PSECT SUMMARY
;
;       Name                        Bytes                          Attributes
;
;   $OWN$                                168  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;   $PLIT$                               148  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;   $CODE$                              1614  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)



;                             Library Statistics
;
;                                          -------- Symbols --------        Pages       Processing
;       File                               Total    Loaded   Percent        Mapped      Time
;
;   _$255$DUA28:[MOM.OBJ]MOMLIB.L32;1       194        49        25            21        00:00.1
;   _$255$DUA28:[SHRLIB]NMALIBRY.L32;1      887        14         1            47        00:00.2
;   _$255$DUA28:[SHRLIB]NET.L32;1          1279        22         1            63        00:00.3
;   _$255$DUA28:[SYSLIB]STARLET.L32;1      9776         3         0           581        00:03.2



;                                      COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:MOMSUBS/OBJ=OBJ$:MOMSUBS MSRC$:MOMSUBS/UPDATE=(ENH$:MOMSUBS)

; Size:           1614 code + 316 data bytes
; Run Time:          00:34.9
; Elapsed Time:      01:12.0
; Lines/CPU Min:       1801
; Lexemes/CPU-Min: 15995
; Memory Used:  211 pages
; Compilation Complete
```

0238 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY